

```

BBBBBBBBBBBBBB      AAAAAAAAAA      SSSSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTTTT      LLL
BBBBBBBBBBBBBB      AAAAAAAAAA      SSSSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTTTT      LLL
BBBBBBBBBBBBBB      AAAAAAAAAA      SSSSSSSSSSSSSS      RRRRRRRRRRRR      TTTTTTTTTTTTTTTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAAAAAAAAAAAAAAAAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAAAAAAAAAAAAAAAAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAAAAAAAAAAAAAAAAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBB      BBB      AAA      AAA      SSS      SSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSSSSSSSSSSSSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSSSSSSSSSSSSS      RRR      RRR      TTT      TTT      LLL
BBBBBBBBBBBBBB      AAA      AAA      SSSSSSSSSSSSSS      RRR      RRR      TTT      TTT      LLL

```

```
BBBBBBBBB      AAAAAA      SSSSSSSS      BBBBBBBB      UU      UU      FFFFFFFF      SSSSSSSS      IIIIII      ZZZZZZZZZZ
BBBBBBBBB      AAAAAA      SSSSSSSS      BBBBBBBB      UU      UU      FFFFFFFF      SSSSSSSS      IIIIII      ZZZZZZZZZZ
BB      BB      AA      AA      SS      BB      BB      UU      UU      FF      SS      II      ZZ
BB      BB      AA      AA      SS      BB      BB      UU      UU      FF      SS      II      ZZ
BB      BB      AA      AA      SS      BB      BB      UU      UU      FF      SS      II      ZZ
BBBBBBBBB      AA      AA      SSSSSS      BBBBBBBB      UU      UU      FFFFFFFF      SSSSSS      II      ZZ
BBBBBBBBB      AA      AA      SSSSSS      BBBBBBBB      UU      UU      FFFFFFFF      SSSSSS      II      ZZ
BB      BB      AAAAAAAAAA      SS      BB      BB      UU      UU      FF      SS      II      ZZ
BB      BB      AAAAAAAAAA      SS      BB      BB      UU      UU      FF      SS      II      ZZ
BB      BB      AA      AA      SS      BB      BB      UU      UU      FF      SS      II      ZZ
BB      BB      AA      AA      SS      BB      BB      UU      UU      FF      SS      II      ZZ
BBBBBBBBB      AA      AA      SSSSSSSS      BBBBBBBB      UUUUUUUUUU      FF      SSSSSSSS      IIIIII      ZZZZZZZZZZ
BBBBBBBBB      AA      AA      SSSSSSSS      BBBBBBBB      UUUUUUUUUU      FF      SSSSSSSS      IIIIII      ZZZZZZZZZZ

LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLL      IIIIII      SSSSSSSS
```

```
0001 0 MODULE BAS$BUFSIZ (
0002 0 IDENT = '1-003'
0003 0 ) =
0004 1 BEGIN
0005 1
0006 1 *****
0007 1 *
0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0010 1 * ALL RIGHTS RESERVED.
0011 1 *
0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0017 1 * TRANSFERRED.
0018 1 *
0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0021 1 * CORPORATION.
0022 1 *
0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0025 1 *
0026 1 *
0027 1 *****
0028 1
0029 1
0030 1 ++
0031 1 FACILITY: VAX-11 BASIC Miscellaneous I/O
0032 1
0033 1 ABSTRACT:
0034 1
0035 1 This module contains the BASIC BUFSIZ function, which returns
0036 1 the buffer size of the file open on the specified channel.
0037 1
0038 1 ENVIRONMENT: VAX-11 User Mode
0039 1
0040 1 AUTHOR: John Sauter, CREATION DATE: 11-APR-1979
0041 1
0042 1 MODIFIED BY:
0043 1
0044 1 1-001 - Original.
0045 1 1-002 - Set up ISB$A_USER_FP. JBS 25-JUL-1979
0046 1 1-003 - Use channel 0, not device TT.
0047 1 JBS 11-MAR-1980
0048 1 --
0049 1
0050 1 !<BLF/PAGE>
```



```
52 0051 1 |
53 0052 1 | SWITCHES:
54 0053 1 |
55 0054 1 |
56 0055 1 | SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
57 0056 1 |
58 0057 1 |
59 0058 1 | LINKAGES:
60 0059 1 |
61 0060 1 |
62 0061 1 | REQUIRE 'RTLIN:OTSLNK'; | Define linkages
63 0490 1 |
64 0491 1 |
65 0492 1 | TABLE OF CONTENTS:
66 0493 1 |
67 0494 1 |
68 0495 1 | FORWARD ROUTINE
69 0496 1 | BASSBUFSIZ; | Return buffer size
70 0497 1 |
71 0498 1 |
72 0499 1 | INCLUDE FILES:
73 0500 1 |
74 0501 1 |
75 0502 1 | REQUIRE 'RTLML:OTSLUB'; | Get LUB definitions
76 0642 1 |
77 0643 1 | REQUIRE 'RTLML:OTSISB'; | Get ISB definitions
78 0811 1 |
79 0812 1 | REQUIRE 'RTLIN:RTLPSECT'; | Macros for defining psects
80 0907 1 |
81 0908 1 | LIBRARY 'RTLSTARLE'; | System symbols
82 0909 1 |
83 0910 1 |
84 0911 1 | MACROS:
85 0912 1 |
86 0913 1 | NONE
87 0914 1 |
88 0915 1 | EQUATED SYMBOLS:
89 0916 1 |
90 0917 1 | NONE
91 0918 1 |
92 0919 1 | PSECTS:
93 0920 1 |
94 0921 1 | DECLARE_PSECTS (BAS); | Declare psects for BASS facility
95 0922 1 |
96 0923 1 | OWN STORAGE:
97 0924 1 |
98 0925 1 | NONE
99 0926 1 |
100 0927 1 | EXTERNAL REFERENCES:
101 0928 1 |
102 0929 1 |
103 0930 1 | EXTERNAL ROUTINE
104 0931 1 | BASS$OPEN ZERO : NOVALUE, | Open channel 0
105 0932 1 | BASS$CB_PUSH : JSB CB_PUSH NOVALUE, | Load register CCB
106 0933 1 | BASS$CB_POP : JSB CB_POP NOVALUE, | Done with register CCB
107 0934 1 | BASS$STOP : NOVALUE; | Signal fatal error
108 0935 1 |
```

BAS\$BUFSIZ
1-003

J 15
16-Sep-1984 00:02:51
14-Sep-1984 11:54:43

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BAS\$BUFSIZ.B32;1

Page 3
(2)

```
: 109      0936 1 !+
: 110      0937 1 !- The following are the error codes used in this module.
: 111      0938 1 !-
: 112      0939 1
: 113      0940 1 EXTERNAL LITERAL
: 114      0941 1     BAS$K_PROLOSSOR : UNSIGNED (8);
: 115      0942 1                                ! Program lost, sorry
```

```
117 0943 1 GLOBAL ROUTINE BASSBUFSIZ (      ! Return buffer size
118 0944 1     CHAN                        ! Channel whose buffer size to return
119 0945 1     ) =
120 0946 1
121 0947 1 ++
122 0948 1 FUNCTIONAL DESCRIPTION:
123 0949 1
124 0950 1     Returns the size of the buffer for the specified channel.
125 0951 1     If the channel is closed a zero is returned.
126 0952 1
127 0953 1 FORMAL PARAMETERS:
128 0954 1
129 0955 1     CHAN.rl.v      The channel whose buffer size to return.
130 0956 1
131 0957 1 IMPLICIT INPUTS:
132 0958 1
133 0959 1     The LUB$W_RBUF_SIZE field of the LUB of the specified channel.
134 0960 1
135 0961 1 IMPLICIT OUTPUTS:
136 0962 1
137 0963 1     NONE
138 0964 1
139 0965 1 ROUTINE VALUE:
140 0966 1
141 0967 1     The number of bytes in the buffer, as a longword integer.
142 0968 1
143 0969 1 SIDE EFFECTS:
144 0970 1
145 0971 1     Signals if an error is encountered.
146 0972 1     BASS$CB_PUSH will signal if the channel number is invalid.
147 0973 1
148 0974 1 --
149 0975 1
150 0976 2 BEGIN
151 0977 2
152 0978 2 BUILTIN
153 0979 2     FP;
154 0980 2
155 0981 2 GLOBAL REGISTER
156 0982 2     CCB = K_CCB_REG : REF BLOCK [, BYTE];
157 0983 2
158 0984 2 LOCAL
159 0985 2     BUFFER_SIZE,
160 0986 2     FMP : REF BLOCK [, BYTE];
161 0987 2
162 0988 2     FMP = .FP;
163 0989 2
164 0990 2 ++
165 0991 2     If this is channel zero, get the user's terminal, and be sure it is open.
166 0992 2
167 0993 2 IF (.CHAN EQL 0)
168 0994 2 THEN
169 0995 2     BEGIN
170 0996 2     BASS$CB_PUSH (LUB$K_LUN_INPU, LUB$K_ILUN_MIN);
171 0997 2     CCB [ISB$A_USER_FP] = .FMP [SF$A_SAVE_FP];
172 0998 2
173 0999 2     IF ( NOT .CCB [LUB$V_OPENED]) THEN BASS$OPEN_ZERO (.FMP [SF$A_SAVE_FP]);
```


BAS\$BUFSIZ
1-003

L 15
16-Sep-1984 00:02:51
14-Sep-1984 11:54:43

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BAS\$BUFSIZ.B32;1

Page 5
(3)

```
174 1000 3
175 1001
176 1002
177 1003
178 1004
179 1005
180 1006
181 1007
182 1008
183 1009
184 1010
185 1011
186 1012
187 1013
188 1014
189 1015
190 1016
191 1017
192 1018
193 1019
194 1020
195 1021

      END
    ELSE
      BEGIN
        BAS$$CB_PUSH (.CHAN, LUB$K_LUN_MIN);
        CCB [ISB$A_USER_FP] = .FMP [SF$L_SAVE_FP];
      END;

      +
      Get the buffer size from the Logical Unit Block. This will be zero
      if the channel has not been opened.
      -
      BUFFER_SIZE = .CCB [LUB$W_RBUF_SIZE];
      +
      We are done with register CCB.
      -
      BAS$$CB_POP ();
      +
      All done.
      -
      RETURN (.BUFFER_SIZE);
    END;
```

! end of BAS\$BUFSIZ

.TITLE BAS\$BUFSIZ
.IDENT \1-003\

.EXTRN BAS\$OPEN_ZERO, BAS\$\$CB_PUSH
.EXTRN BAS\$\$CB_POP, BAS\$\$STOP
.EXTRN BAS\$K_PROLOSSOR

.PSECT _BAS\$CODE, NOWRT, SHR, PIC, 2

.ENTRY BAS\$BUFSIZ, Save R2,R3,R4,R11
MOVAB BAS\$\$CB_PUSH, R4
MOVL FP, FMP
TSTL CHAN
BNEQ 1\$
MNEGL #8, R0
MNEGL #7, R2
JSB BAS\$\$CB_PUSH
MOVL 12(FMP), -180(CCB)
BLBS -4(CCB), 2\$
PUSHL 12(FMP)
CALLS #1, BAS\$OPEN_ZERO
BRB 2\$
CLRL R0
MOVL CHAN, R2
JSB BAS\$\$CB_PUSH
MOVL 12(FMP), -180(CCB)
MOVZWL -4(CCB), BUFFER_SIZE
JSB BAS\$\$CB_POP
MOVL BUFFER_SIZE, R0
RET

```
081C 00000
54 00000000G 00 9E 00002
53          5D D0 00009
          04 AC D5 0000C
          1E 12 0000F
50          08 CE 00011
52          07 CE 00014
          64 16 00017
FF4C CB 0C A3 D0 00019
1A FC AB E8 0001F
          0C A3 DD 00023
00000000G 00 01 FB 00026
          0E 11 0002D
          50 D4 0002F 1$:
52          04 AC D0 00031
          64 16 00035
FF4C CB 0C A3 D0 00037
52 D2 AB 3C 0003D 2$:
          00000000G 00 16 00041
50          52 D0 00047
          04 0004A
```

; Routine Size: 75 bytes, Routine Base: _BAS\$CODE + 0000

0943
0988
0993
0996
0997
0999
0993
1004
1005
1012
1016
1020
1021

BAS\$BUFSIZ
1-003

M 15
16-Sep-1984 00:02:51
14-Sep-1984 11:54:43

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BAS\$BUFSIZ.B32:1

Page 6
(3)

```
: 196      1022 1
: 197      1023 1 END
: 198      1024 1
: 199      1025 0 ELUDOM
```

! end of module BAS\$BUFSIZ

PSECT SUMMARY

Name	Bytes	Attributes
_BAS\$CODE	75	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32:1	9776	1	0	581	00:01.2

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LIS\$:BAS\$BUFSIZ/OBJ=OBJ\$:BAS\$BUFSIZ MSRC\$:BAS\$BUFSIZ/UPDATE=(ENH\$:BAS\$BUFSIZ)

```
: Size:      75 code + 0 data bytes
: Run Time:   00:08.3
: Elapsed Time: 00:20.5
: Lines/CPU Min: 7436
: Lexemes/CPU-Min: 45155
: Memory Used: 115 pages
: Compilation Complete
```


0019

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY